**Задача 1:**

Вам необходимо создать таблицу для хранения информации о студентах в университете. Таблица должна содержать следующие поля: - Идентификатор студента (целочисленный тип данных) - Имя студента (строковый тип данных) - Фамилия студента (строковый тип данных) - Год рождения студента (дата тип данных) - Средний балл студента (вещественный тип данных) Создайте SQL запрос для создания такой таблицы

CREATE TABLE students (

`id` int unsigned NOT NULL AUTO\_INCREMENT,

`name` varchar(45) NOT NULL,

`surname` varchar(45) NOT NULL,

`year\_birth` date NOT NULL,

`gpa` decimal(5,2) unsigned DEFAULT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB

DEFAULT CHARSET=utf8

**Задача 2:**

Вам необходимо создать таблицу для хранения информации о книгах в библиотеке. Таблица должна содержать следующие поля: - Идентификатор книги (целочисленный тип данных, уникальный идентификатор книги) - Название книги (строковый тип данных) - Автор книги (строковый тип данных) - Год издания книги (целочисленный тип данных) - Жанр книги (строковый тип данных) Убедитесь, что поле "Идентификатор книги" является первичным ключом (PRIMARY KEY). Создайте SQL запрос для создания такой таблицы.

CREATE TABLE first\_db.books (

`id` INT NOT NULL,

`booktitle` VARCHAR(45) NULL,

`author` VARCHAR(45) NULL,

`year\_publishing` INT NULL,

`genre` VARCHAR(45) NULL,

PRIMARY KEY (`id`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8;

ALTER TABLE `first\_db`.`books`

CHANGE COLUMN `id` `id` INT UNSIGNED NOT NULL AUTO\_INCREMENT ,

CHANGE COLUMN `booktitle` `booktitle` VARCHAR(45) NOT NULL ,

CHANGE COLUMN `author` `author` VARCHAR(45) NOT NULL ,

CHANGE COLUMN `year\_publishing` `year\_publishing` INT UNSIGNED NOT NULL ,

CHANGE COLUMN `genre` `genre` VARCHAR(45) NOT NULL ;

INSERT INTO `first\_db`.`books` (`id`, `booktitle`, `author`, `year\_publishing`, `genre`) VALUES ('1', 'Гордость и предубеждение', 'Джейн Остин', '1813', ' роман');

INSERT INTO `first\_db`.`books` (`id`, `booktitle`, `author`, `year\_publishing`, `genre`) VALUES ('2', 'Воскресение', 'Лев Толстой', '1899', 'драма');

INSERT INTO `first\_db`.`books` (`id`, `booktitle`, `author`, `year\_publishing`, `genre`) VALUES ('3', 'Грозовой перевал', 'Эмили Бронте', '1847', 'роман');

**3. Задача:**

Вы работаете в книжном магазине и вам необходимо добавить новую книгу в базу данных. Вам известны следующие данные о книге: - Название: "Повелитель мух" - Автор: Уильям Голдинг - Год издания: 1954 - Жанр: Роман Составьте SQL запрос для добавления этой книги в таблицу "books" в вашей базе данных.

INSERT INTO first\_db.books (`booktitle`, `author`, `year\_publishing`, `genre`) VALUES ('Повелитель мух', 'Уильям Голдинг', '1954', 'роман');

**4. Задача:**

Вы работаете в компании, которая занимается продажей электроники, и вам необходимо обновить информацию о ценах на определенные товары. Вам известны следующие данные: - Таблица "products" содержит информацию о товарах, включая поля: product\_id (идентификатор товара), product\_name (название товара), price (цена товара). - Вам нужно увеличить цену на 10% для всех товаров, у которых цена ниже 1000 рублей. Составьте SQL запрос для обновления цен на 10% для соответствующих товаров в таблице "products"

CREATE TABLE `first\_db`.`products` (

`product\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

`product\_name` VARCHAR(45) NOT NULL,

`price` INT NOT NULL,

PRIMARY KEY (`product\_id`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8;

INSERT INTO `first\_db`.`products` (`product\_id`, `product\_name`, `price`) VALUES ('1', 'ноутбук', '1500');

INSERT INTO `first\_db`.`products` (`product\_id`, `product\_name`, `price`) VALUES ('2', 'принтер', '1000');

INSERT INTO `first\_db`.`products` (`product\_id`, `product\_name`, `price`) VALUES ('3', 'телевизор', '900');

INSERT INTO `first\_db`.`products` (`product\_id`, `product\_name`, `price`) VALUES ('4', 'телефон', '300');

**SET SQL\_SAFE\_UPDATES = 0;**

UPDATE first\_db.products SET `price` = price\*1.1 WHERE (`price` < 1000);

**5. Задача:**

Вам необходимо вывести информацию о самых дорогих товарах в вашем магазине. У

вас есть таблица "products" с полями: product\_id (идентификатор товара), product\_name

(название товара) и price (цена товара). Напишите SQL запрос, который выведет

названия и цены трех самых дорогих товаров из таблицы "products".

SELECT product\_name,

price

FROM first\_db.products

ORDER BY price DESC

LIMIT 3;

**6. Задача:**

В вашей базе данных есть таблица "employees" с полями: employee\_id

(идентификатор сотрудника), employee\_name (имя сотрудника) и department\_id

(идентификатор отдела, к которому принадлежит сотрудник). Вам необходимо удалить

всех сотрудников, которые работают в отделе с department\_id = 3. Напишите SQL

запрос для выполнения этой операции.

CREATE TABLE first\_db.employees (

`employee\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

`employee\_name` VARCHAR(50) NOT NULL,

`department\_id` INT NOT NULL,

PRIMARY KEY (`employee\_id`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8;

INSERT INTO first\_db.employees (`employee\_id`, `employee\_name`, `department\_id`) VALUES (1, 'Evgeniya', 1);

INSERT INTO first\_db.employees (`employee\_id`, `employee\_name`, `department\_id`) VALUES (2, 'Stepa', 2);

INSERT INTO first\_db.employees (`employee\_id`, `employee\_name`, `department\_id`) VALUES (3, 'Misha', 3);

INSERT INTO first\_db.employees (`employee\_id`, `employee\_name`, `department\_id`) VALUES (4, 'Alex', 4);

INSERT INTO first\_db.employees (`employee\_id`, `employee\_name`, `department\_id`) VALUES (5, 'Mary', 3);

SET SQL\_SAFE\_UPDATES = 0;

DELETE FROM first\_db.employees WHERE (`department\_id` = '3');

**7. Задача:**

У вас есть две таблицы: "students" с полями student\_id (идентификатор студента) и student\_name (имя студента) и "courses" с полями course\_id (идентификатор курса) и course\_name (название курса). Необходимо создать внешний ключ (foreign key) в таблице "students", который будет ссылаться на поле course\_id из таблицы "courses". При этом необходимо убедиться, что при удалении курса из таблицы "courses", все записи о студентах, которые выбрали этот курс, также будут удалены. Напишите SQL запрос для создания внешнего ключа с таким поведением.

CREATE TABLE first\_db.students1 (

`student\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

`student\_name` VARCHAR(50) NOT NULL,

PRIMARY KEY (`student\_id`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8;

CREATE TABLE first\_db.courses1 (

`course\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

`curse\_name` VARCHAR(50) NOT NULL,

PRIMARY KEY (`course\_id`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8;

ALTER TABLE first\_db.students1

ADD `course\_id` INT UNSIGNED NOT NULL;

ALTER TABLE first\_db.students1

ADD CONSTRAINT FK\_CourseId

FOREIGN KEY (course\_id) REFERENCES courses1(course\_id)

ON DELETE CASCADE;

**8. Задача:**

У вас есть две таблицы: "отделы" (departments) с полями department\_id (идентификатор отдела) и department\_name (название отдела) и "сотрудники" (employees) с полями employee\_id (идентификатор сотрудника), employee\_name (имя сотрудника) и department\_id (идентификатор отдела, к которому принадлежит сотрудник). Необходимо вывести список всех отделов и количество сотрудников в каждом отделе. Напишите SQL запрос для решения этой задачи.

CREATE TABLE first\_db.departments (

`iddepartment\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

`department\_name` VARCHAR(45) NOT NULL,

PRIMARY KEY (`department\_id`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8;

INSERT INTO first\_db.departments (`department\_id`, `department\_name`) VALUES (1, 'marketing');

INSERT INTO first\_db.departments (`department\_id`, `department\_name`) VALUES (2, 'sales');

INSERT INTO first\_db.departments (`department\_id`, `department\_name`) VALUES (3, 'head');

INSERT INTO first\_db.departments (`department\_id`, `department\_name`) VALUES (4, 'connection');

ALTER TABLE `first\_db`.`employees`

CHANGE COLUMN `department\_id` `department\_id` INT UNSIGNED NOT NULL ;

ALTER TABLE first\_db.employees

ADD CONSTRAINT `department\_id`

FOREIGN KEY (`department\_id`)

REFERENCES first\_db.departments (`department\_id`);

INSERT INTO first\_db.employees (employee\_name, department\_id) VALUES ('Sveta', 1);

INSERT INTO first\_db.employees (employee\_name, department\_id) VALUES ('Marysya', 4);

INSERT INTO first\_db.employees (employee\_name, department\_id) VALUES ('Sergeq', 4);

INSERT INTO first\_db.employees (employee\_name, department\_id) VALUES ('Nina', 3);

SELECT COUNT(first\_db.departments.department\_id), first\_db.departments.department\_name

FROM first\_db.departments

INNER JOIN first\_db.employees ON first\_db.departments.department\_id=first\_db.employees.department\_id

GROUP BY first\_db.departments.department\_id;

**9. Задача:**

У вас есть две таблицы: "студенты" (students) с полями student\_id (идентификатор студента), student\_name (имя студента) и group\_id (идентификатор группы, к которой принадлежит студент) и "группы" (groups) с полями group\_id (идентификатор группы) и group\_name (название группы). Необходимо вывести список всех студентов и название группы, к которой они принадлежат. Используйте операцию INNER JOIN для объединения таблиц. Напишите SQL запрос для решения этой задачи.

CREATE TABLE first\_db.students2 (

`student\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

`student\_name` VARCHAR(50) NOT NULL,

`group\_id` INT NOT NULL,

PRIMARY KEY (`student\_id`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8;

CREATE TABLE first\_db.groups (

`group\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

`group\_name` VARCHAR(50) NOT NULL,

PRIMARY KEY (`group\_id`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8;

ALTER TABLE first\_db.students2

CHANGE COLUMN `group\_id` `group\_id` INT UNSIGNED NOT NULL ;

ALTER TABLE first\_db.students2

ADD CONSTRAINT FK\_GroupId

FOREIGN KEY (group\_id) REFERENCES first\_db.groups(group\_id);

INSERT INTO first\_db.groups (`group\_id`, `group\_name`) VALUES (1, 'Programming');

INSERT INTO first\_db.groups (`group\_id`, `group\_name`) VALUES (2, 'Management');

INSERT INTO first\_db.groups (`group\_id`, `group\_name`) VALUES (3, 'Economy');

INSERT INTO first\_db.students2 (`student\_id`, `student\_name`, `group\_id`) VALUES (1, 'Eva', 1);

INSERT INTO first\_db.students2 (`student\_id`, `student\_name`, `group\_id`) VALUES (2, 'Sasha', 1);

INSERT INTO first\_db.students2 (`student\_id`, `student\_name`, `group\_id`) VALUES (3, 'Olya', 2);

INSERT INTO first\_db.students2 (`student\_id`, `student\_name`, `group\_id`) VALUES (4, 'Veronika', 3);

INSERT INTO first\_db.students2 (`student\_id`, `student\_name`, `group\_id`) VALUES (5, 'Stepan', 3);

INSERT INTO first\_db.students2 (`student\_id`, `student\_name`, `group\_id`) VALUES (6, 'Misha', 2);

INSERT INTO first\_db.students2 (`student\_id`, `student\_name`, `group\_id`) VALUES (7, 'Alex', 1);

INSERT INTO first\_db.students2 (`student\_id`, `student\_name`, `group\_id`) VALUES (8, 'Vika', 2);

INSERT INTO first\_db.students2 (`student\_id`, `student\_name`, `group\_id`) VALUES (9, 'Sveta', 3);

SELECT first\_db.students2.student\_name, first\_db.groups.group\_name

FROM first\_db.students2

INNER JOIN first\_db.groups ON first\_db.students2.group\_id=first\_db.groups.group\_id;

**10. Задача:** У вас есть две таблицы: "авторы" (authors) с полями author\_id (идентификатор автора) и author\_name (имя автора) и "книги" (books) с полями book\_id (идентификатор книги), book\_title (название книги) и author\_id (идентификатор автора книги). Одному автору может принадлежать несколько книг, и одна книга может иметь несколько авторов. Необходимо вывести список всех книг с указанием всех их авторов. Используйте операцию INNER JOIN для объединения таблиц. Напишите SQL запрос для решения этой задачи.

CREATE TABLE first\_db.books1 (

`book\_id ` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

`book\_title` VARCHAR(225) NOT NULL,

`author\_id` INT UNSIGNED NOT NULL,

PRIMARY KEY (`book\_id`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8;

CREATE TABLE first\_db.authors (

`author\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

`author\_name` VARCHAR(225) NOT NULL,

PRIMARY KEY (`author\_id`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8;

ALTER TABLE first\_db.books1

ADD CONSTRAINT FK\_AuthorId

FOREIGN KEY (author\_id ) REFERENCES first\_db.authors(author\_id );

INSERT INTO first\_db.authors VALUES (1, 'Джейн Остин');

INSERT INTO first\_db.authors VALUES (2, 'Илья Ильф');

INSERT INTO first\_db.authors VALUES (3, 'Евгений Петров');

INSERT INTO first\_db.authors VALUES (4, 'Лев Толстой');

INSERT INTO first\_db.authors VALUES (5, 'Эмили Бронте');

INSERT INTO first\_db.books1 VALUES (1, 'Гордость и предубеждение', 1);

INSERT INTO first\_db.books1 VALUES (2, 'Двеннадцать стульев', 2);

INSERT INTO first\_db.books1 VALUES (3, 'Двеннадцать стульев', 3);

INSERT INTO first\_db.books1 VALUES (4, 'Шипы и розы', 2);

INSERT INTO first\_db.books1 VALUES (5, 'Воскресение', 4);

INSERT INTO first\_db.books1 VALUES (6, 'Война и Мир', 4);

INSERT INTO first\_db.books1 VALUES (7, 'Грозовой перевал', 5);

INSERT INTO first\_db.books1 VALUES (8, 'Доводы рассудка', 1);

CREATE TABLE first\_db.author\_Books1 (

`id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

`author\_id` INT UNSIGNED NOT NULL,

`book\_id` INT UNSIGNED NOT NULL,

PRIMARY KEY (`id`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8;

ALTER TABLE first\_db.books1

DROP FOREIGN KEY `FK\_AuthorId`;

ALTER TABLE `first\_db`.`books1`

DROP COLUMN `author\_id`,

DROP INDEX `FK\_AuthorId` ;

;

INSERT INTO first\_db.author\_books1 (`author\_id`, `book\_id`) VALUES (1, 1);

INSERT INTO first\_db.author\_books1 (`author\_id`, `book\_id`) VALUES (1, 8);

INSERT INTO first\_db.author\_books1 (`author\_id`, `book\_id`) VALUES (2, 2);

INSERT INTO first\_db.author\_books1 (`author\_id`, `book\_id`) VALUES (2, 4);

INSERT INTO first\_db.author\_books1 (`author\_id`, `book\_id`) VALUES (3, 3);

INSERT INTO first\_db.author\_books1 (`author\_id`, `book\_id`) VALUES (4, 5);

INSERT INTO first\_db.author\_books1 (`author\_id`, `book\_id`) VALUES (4, 6);

INSERT INTO first\_db.author\_books1 (`author\_id`, `book\_id`) VALUES (5, 7);

ALTER TABLE first\_db.author\_books1

ADD CONSTRAINT FK\_Authorid

FOREIGN KEY (author\_id)

REFERENCES first\_db.authors (author\_id);

ALTER TABLE first\_db.author\_books1

ADD CONSTRAINT FK\_Bookid

FOREIGN KEY (book\_id)

REFERENCES first\_db.books1 (book\_id)

SELECT first\_db.authors.author\_name, first\_db.books1.book\_title

FROM first\_db.books1

INNER JOIN first\_db.author\_books1

ON first\_db.books1.book\_id = first\_db.author\_books1.book\_id

INNER JOIN first\_db.authors

ON first\_db.authors.author\_id = first\_db.author\_books1.author\_id;